第 19 卷第 4 期　　半 导 体 学 报　　Vol. 19, No. 4
1998 年 4 月　　CHINESE JOURNAL OF SEMICONDUCTORS　　Apr., 1998

性能驱动系统划分的均场退火方法*

胡卫明　何厚武　严晓浪

(杭州电子工业学院 CAD 所　杭州　310037)

摘要　本文通过把时延约束条件转化为一个布尔矩阵, 利用这个布尔矩阵构造能量函数的时延约束项, 从而解决了用神经网络处理时延约束这类不等式约束的难题, 并据此提出了一个性能驱动的 VLSI 系统划分的均场退火算法. 算法不仅考虑了模块间的连接关系, 还考虑了版图的物理结构, 是一种逻辑与版图相结合的划分方法. 实验表明, 该算法具有较强的寻优能力.

CCACC: 7410D, 5120

1　引言

对于由相互连接的模块构成的电路或系统, 主要有两类划分问题. 一类是块与块之间没有特定的拓扑关系, 因此可用"Ratio-Cut"代价函数作目标, 它用于决定电路的结构和寻找电路的所谓"自然群集". 另一类的块与块之间具有确定的拓扑关系, 包括每块的容量、块与块之间的连线代价和时延模型等, FPGA 和一些 MCM 类型的划分即属此类.

本文讨论的是第二类划分问题. 模块的面积是指实现这一模块所要占用的硅面积, 而每个划分块的确定的硅面积构成了块的面积容量. 面积约束是指分配到同一划分块的模块的面积总和不能大于该划分块的面积. 时延约束是由系统周期时间驱动的, 并且可以通过时延方程和组合电路模块的内部时延推导得出两个模块间所允许的最大布线时延[1].

划分是指把一个系统分割成若干子系统, 使得每个子系统不太复杂, 从而便于实现. 围绕划分问题, 多年来提出了大量的启发式算法. 根据初始划分的实用性, 可分为构造划分算法和迭代划分算法两类. 根据算法的性质有确定性划分算法和随机性划分算法两类. 构造划分方法主要有结群、基于布局的划分、数学规划和网络流计算等方法; 最近随机构造算法也用来解决划分问题[2]; 大量的则是确定性迭代改进方法, 如最小割算法; 随机迭代改进方法

* 国家"九五"重点科技攻关资助项目
胡卫明　男, 1968 年出生, 博士研究生, 从事 ICCAD 和人工神经网络的研究工作
何厚武　男, 1970 年出生, 硕士研究生, 从事集成电路布局布线的研究工作
严晓浪　男, 1946 年出生, 博士导师, 从事 ICCAD 和设计自动化领域的教学与科研
1997-01-12 收到, 1997-06-14 定稿

主要有模拟退火算法和模拟进化算法. 这些算法本质都是串行的, 不能有效地组织成并行结构. 神经网络作为一种新的有并行结构的优化方法, 也有人尝试用来解决电路划分问题[4, 5].

目前, 神经网络还只局限于解决面积驱动的电路划分问题, 这是因为如同时延约束这种类型的不等式约束不能直接加入到神经网络的能量函数中. 本文将时延约束条件转化为一个布尔矩阵, 利用这个布尔矩阵建立了能量函数的时延约束项, 从而提出了一个性能驱动的系统划分算法. 算法应用均场退火神经网络来求解以减小块间的连线代价为目标、以面积约束和时延约束为约束条件的优化问题. 算法中, 每个模块只能分配到一个划分块中的约束(简称"1 of N"约束)用神经元归一化的方法解决.

2　问题描述

2.1　实例描述

一个划分实例包括如下内容[1]:

2.1.1　电路描述:

1. I 是由 M 个模块组成的电路模块集合. $i \in I$ 是一个模块索引.
2. s_i 是模块 i 的面积.
3. A 是一个 $M \times M$ 矩阵, 其中 a_{ij} 是从模块 i 到模块 j 的连线数.
4. C 是一个 $M \times M$ 矩阵. 其中 c_{ij} 是从模块 i 到模块 j 所允许的最大信号布线时延.

2.1.2　划分块描述:

1. X 是由 N 个划分块组成的划分块集合. $x \in X$ 是一个块的索引.
2. t_x 为块 x 的面积容量.
3. B 是一个 $N \times N$ 矩阵, 其中 b_{xy} 是从块 x 到块 y 的布线代价, 它可以是块 x 到块 y 的 Manhattan 距离, 或其它形式.
4. D 是一个 $N \times N$ 矩阵, 其中 d_{xy} 是从块 x 到块 y 的布线时延. 为了问题的定义更具一般性, B 和 D 的相互关系未被定义.

2.2　问题描述

性能驱动的系统划分问题可以描述为: 给定一个划分实例, 要寻求一个分配 $\psi: I \to X$, 使 $\sum_{\forall i,j} a_{ij} b_{\psi(i),\psi(j)}$ 最小, 并满足如下几个约束集合:

$C1$(容量约束): $\sum_{\forall i \psi(i)=x} s_i \le t_x \quad \forall x \in X$

$C2$(时延约束): $D(\psi(i), \psi(j)) \le C(i, j) \quad \forall i, j \in I$

$C3$("1 of N"约束): $x \cap y = \Phi, x \ne y \quad \forall x, y \in X$

3　均场退火网络

模拟退火算法是一个通用的、与领域无关的、具有概率爬山性的组合优化算法. 但由于它要求温度下降足够慢, 因而导致了该算法的计算时间很长. 均场退火既可看作是一种新的神经网络计算模型, 又可视为是对模拟退火的重大改进. 它只需要在某个关键温度附近实施退火过程就可取得较好的效果, 因此计算时间大为减少[6].

均场退火可由下述方程描述:

$$\begin{cases} E = f(V) \\ \varphi_i = \dfrac{\partial E}{\partial v_i} = \dfrac{\partial f}{\partial v_i} \\ v_i = g(\varphi_i, T) \end{cases}$$

其中 V 是能量函数中的 N 维状态矢量; φ 为均场; g 是神经元的 I/O 特性函数; T 是温度控制参数. 所谓均场退火就是在某一个关键温度 T_c 附近实施模拟退火过程, 以达到某个热平衡状态, 这样即可得到一个最优解或较好的次优解.

4 主要算法

4.1 模块的归并

在进行神经计算之前, 可以根据时延约束条件, 对模块进行归并处理, 以减少模块的数量, 从而缩小问题的规模, 减少实施均场退火所需的计算时间. 归并的方法为: 对于任意两个模块 i 和 j, 如果 $C(i,j) < \min_{x,y}(D(x,y))$, 则归并模块 i 和模块 j 为模块 k.

4.2 映射方法

将 M 个模块的 N 块划分问题映射为神经网络, 需要用换位矩阵 $[v_{ix}]_{M\times N}$ 表示: 矩阵的行代表模块, 列代表所属的划分块. 换位矩阵是一个值为"0"或"1"的矩阵, 它对应于一种可行的划分. 该问题需要 $M \times N$ 个神经元, 如图 1 所示. 当 $\psi(i) = x$, 即模块 i 应属于划分块 x, 神经元 (i,x) 的输出 $v_{ix} = 1$, 否则 $v_{ix} = 0$.

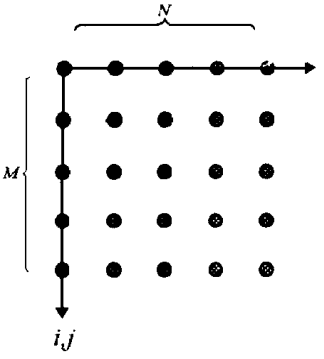

图 1 映射方法

4.3 能量函数

在能量函数的定义中, 除了定义目标项以外, 还要定义面积约束项和时延约束项.

4.3.1 目标项

将问题映射相应的神经网络后, 目标函数转化为:

$$E_1 = \sum_{i=1}^{M}\sum_{x=1}^{N}\sum_{j=1}^{M}\sum_{y=1}^{N} a_{ij} b_{xy} v_{ix} v_{jy}$$

它与模块间的连线代价相联系, 可用于模型化任何形式的连线代价矩阵. 例如, 当 B 的主对角线元素为"0", 其余元素为"1"时, 这一项对应于某一分配 ψ 的块间连线总数; 当 b_{xy} 为块 x 到块 y 的 Manhattan 距离, 该项等价于块间连线的 Manhattan 距离总长. 可以根据问题的需要, 定义代价函数 B 的形式.

4.3.2 面积约束项

面积约束项的定义考虑了以下三种情况:

1) 若各个划分块的面积相同, 各个模块的面积也相同, 则面积约束项可以定义为:

$$E_2 = -\frac{\alpha}{2}\sum_{i=1}^{M}\sum_{j\neq i}^{M}\sum_{x=1}^{N} v_{ix}(1 - v_{jx})$$

其含义为当模块 i 和 j 分配在不同的块时, 加入一个负能量.

2) 若各个划分块的面积相同, 但各个模块的面积不同, 则面积约束可以定义为:

$$E_2 = \frac{\alpha}{2}\sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{x=1}^{N} s_i s_j v_{ix} v_{jx}$$

它表示 $\alpha\sum_{x=1}^{N}(\sum_{i\in S_x} s_i)^2$, 即只有当各划分块中模块面积和相等时它才取最小值, 它利用算术平均和几何平均的关系表示了各划分块中模块面积和大致相等时取极小的约束关系.

3) 若各个划分块的面积不相同, 且各个模块的面积也不相同, 则面积约束项可以定义为:

$$E_2 = \frac{\alpha}{2}\sum_{x=1}^{N}(t_x - \sum_{i=1}^{M} s_i v_{ix})^2$$

4.3.3 时延约束项

如前所述, 神经网络不能直接处理时延约束类型的不等式约束, 为此, 我们定义了一个布尔矩阵 Delay (i: 1～M, j: 1～M, x: 1～N, y: 1～N), 并将时延约束条件转换到该布尔矩阵中. Delay $(i, j, x, y) = 1$ 表示模块 i 和模块 j 能够同时分别分配在块 x 和块 y 中; Delay $(i, j, x, y) = 0$ 表示模块 i 和模块 j 不能同时分别分配到块 x 和块 y 中. Delay 可以由如下程序产生:

if $C(i, j) \geq D(x, y)$ then Delay$(i, j, x, y) = 1$; else Delay$(i, j, x, y) = 0$; endif

有了 Delay 矩阵后, 就可以定义能量函数的时延约束项, 即

$$E_3 = \frac{\beta}{2}\sum_{i=1}^{M}\sum_{x=1}^{N}\sum_{j=1}^{M}\sum_{y=1}^{N}(1 - \mathrm{Delay}(i, j, x, y))\, v_{ix} v_{jy}$$

当 Delay$(i, j, x, y) = 0$ 时, $v_{ix} v_{jy}$ 不能同时为 1, 否则此项不为零.

值得注意的是, Delay 矩阵的定义只是为了说明问题的方便而已, 事实上并无必要用 $O(M^2)$ ($N \ll M$) 数量级的存贮空间去存贮 Delay 矩阵. 程序设计时, 是通过在线计算的方法来求 Delay 值的. 例如, 求 E_3 的方法为: if $C(i, j) < D(x, y)$ then $E_3 = E_3 + 0.5\beta v_{ix} v_{iy}$ endif.

4.3.4 能量函数和均场

本问题的能量函数定义为:

$$\begin{aligned} E &= E_1 + E_2 + E_3 \\ &= \sum_{i=1}^{M}\sum_{x=1}^{N}\sum_{j=1}^{M}\sum_{y=1}^{N} a_{ij} b_{xy} v_{ix} v_{jy} + \frac{\alpha}{2}\sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{x=1}^{N} s_i s_j v_{ix} v_{jx} \\ &\quad + \frac{\beta}{2}\sum_{i=1}^{M}\sum_{x=1}^{N}\sum_{j=1}^{M}\sum_{y=1}^{N}(1 - \mathrm{Delay}(i, j, x, y))\, v_{ix} v_{jy} \end{aligned}$$

由于时延约束比面积约束重要, 所以能量函数中的参数应满足: $\beta > \alpha > 1$. 均场为:

$$\varphi_{ix} = \sum_{j=1}^{M}\sum_{y=1}^{N} a_{ij} b_{xy} v_{jy} + \frac{\alpha}{2}\sum_{j=1}^{M} s_i s_j v_{jx} + \frac{\beta}{2}\sum_{j=1}^{M}\sum_{y=1}^{N}(1 - \mathrm{Delay}(i, j, x, y))\, v_{jy} \qquad (1)$$

4.4 "1 of N" 约束

对于"1 of N"约束, 即 N 个神经元中只有 1 个神经元为"1", 在 Hopfield 网络中是通过"柔软"地往能量函数中加惩罚项的方法解决的. 这种办法外加了自由度, 增加了新的对应不可行解的局部极小点, 从而降低了能量函数向前看的能力. 在均场退火网络中, 可以采用归一化的方法来解决"1 of N"约束.

神经元的输出状态矢量可看作是当模块在随机平衡扰动中模块 i 分配到块 x 的概率, 它服从 Boltzmann 分布, 即 $v_{ix} \approx \exp(-\varphi_{ix}/T)$. 因此, 均场 φ_{ix} 越大, 位置占有概率 v_{ix} 就越小,

这表明模块 i 不可能分配在块 x. 为了获得实际概率, 对神经元的输出 v_{ix} 进行归一化:

$$v_{ix} = \frac{\exp(-\varphi_{ix}/T)}{\sum_{y=1}^{N}\exp(-\varphi_{iy}/T)}$$

它保证了每个模块只能分配在一个划分块中, 从而满足"1 of N "约束. 在高温的条件下, 模块分配到各个划分块的概率呈均匀分布; 而在低温时, 模块向具有较小均值的划分块凝结, 它使整个目标函数最小. 能量函数中的面积约束项、时延约束项和归一化处理保证产生合法解.

4.5 关键温度的确定

欲求临界温度 T_c, 可以认为此时每个模块对各个划分块的占有概率为 $1/N$, 对于每一个神经元 (i, x) 有

$$v_{ix}^0 = 1/N,\quad \varphi_{ix}^0 = \varphi_{ix}(v_{ix} = 1/N)$$

因为 $v_{ix}^0 = \exp(-\varphi_{ix}^0/T_{ix}^0)$

所以 $T_{ix}^0 = -\varphi_{ix}^0/\ln(v_{ix}^0) = \varphi_{ix}^0/\ln(N)$, 选择 $T_c = \max_{i,x}\{T_{ix}^0\}$ 作为整个系统的临界温度, 并在其附近实施退火过程.

4.6 算法描述

根据均场退火方程, 可以将性能驱动系统划分的均场退火过程描述如下:

```
Algorithm MFN
{
    归并模块;
    T = max_{i,x}{T_ix^0}; v_ix = 1/N; //赋初值
        repeat
            随机选择一模块 i; total = 0;
            for x = 1 to N {
                由式(1)计算 φ_x;
                Total = Total + exp(-φ_x/T);
            for x = 1 to N  v_ix = exp(-φ_x/T)/Total;   //归一化处理
            计算 ΔE;
        utill (ΔE ≈ 0)
}
```

说明如下:

1) ΔE 是相邻两次 E 的差. 为了降低求 ΔE 的时间复杂性, 可以用如下方法求 ΔE:

$$\Delta E = \sum_{x=1}^{N}\sum_{j=1}^{M}\sum_{y=1}^{N} a_{ij}b_{xy}(v'_{ix} - v_{ix})v_{jy} + \frac{\alpha}{2}\sum_{j=1}^{M}\sum_{x=1}^{N} s_i s_j (v'_{ix} - v_{ix})v_{jx} + \frac{\beta}{2}\sum_{x=1}^{N}\sum_{j=1}^{M}\sum_{y=1}^{N}(1 - \mathrm{Delay}(i,j,x,y))(v'_{ix} - v_{ix})v_{jy}$$

其中 v_{ix} 和 v'_{ix} 是神经元 (i, x) 的相邻两次输出状态.

2) $\Delta E = 0$ 是热平衡条件.

3) 均场退火理论证明, 只要在关键温度 T_c 附近实施上述的均场退火过程, 即可使 E 最

小或近似最小.

4) 由于$N \ll M$, , 求Φ_x和ΔE的时间复杂性都为$O(M)$; 为了使$\Delta E \to 0$, 需要进行$O(M)$次迭代, 所以本算法的时间复杂性为$O(M^2)$.

5 实验与结论

本算法已用C 语言编程, 在SUN SPARC 2 工作站上实现. 我们首先用图2 这一类的电路对算法进行验证, 图中线的粗细表示模块间连线数目的多少, 图中虚线表示划分的最优解. 经测试, 对于这类例子, 800 个模块划分成4 块时, 可以求得最优解. 这表明本算法有很强的寻优能力.

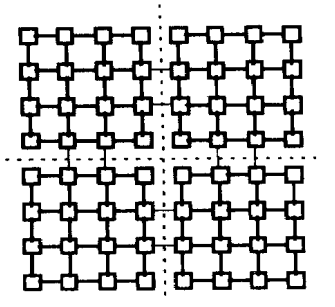

图2　测试实例

另外4 个实际例子也用来对本算法(MFN)进行测试, 并且将它与Hopfield 网络算法(HNN)进行了比较. 表1 列出了这4 个电路的有关信息和这些电路划分成5 个划分块的实验结果. 实验结果表明,MFN 算法的运行速度不如HNN 算法; 但是, 如果MFN 算法用4 倍于HNN 算法的运行时间, 可以取得比HNN 好得多的运行结果. 另外,MFN 算法也比HNN 算法稳定.

表1　实验结果

例号	模块数目	连线数	Manhattam 布线代价总和		运行时间	
			MFN 算法	HNN 算法	MFN 算法	HNN 算法
1	38	173	243	309	0.98	0.17
2	296	3091	6872	7102	106.4	39.7
3	575	7307	13110	18484	319.3	90.2
4	1043	9355	19068	-	859.7	-

注: 例4 算法HNN 迭代发散, 未求得计算结果.

本文应用均场退火网络解决了性能驱动的系统划分问题, 算法不仅考虑了模块间的连接关系, 还考虑了版图的物理结构, 是一种逻辑与版图相结合的划分方法. 算法公式规范简明, 质量高, 具有良好的应用前景.

参　考　文　献

[1] M. Shih and E. S. Kuh, Proc. of DAC, 1993, 761～765.

[2] F. M. Johannes, Proc. of DAC, 1996, 83～86.

[3] C. W. Yok, C. K. Cheng and T. T. Y. Lin, IEEE Trans. on CAD, 1994, **13**(12): 1480～1487.

[4] J. S. Yih and P. Mazumder, IEEE Trans. on CAD, 1990, **9**(12): 1265～1271.

[5] A. Koenig, N. Wehn and M. Glesnet, Proc. of ISCAS, 1992, 324～327.

[6] 焦李成, 神经网络计算, 西安: 西安电子科技大学出版社, 1995.

Mean Field Annealing Approach for Performance-Driven System Partitioning

Hu Weiming, He Houwu and Yan Xiaolang

(*Hangzhou Institute of Electronics Engineering, Hangzhou* 310037)

Received 12 January 1997, revised manuscript received 14 June 1997

Abstract Timing constraints are transformed to a Boolean matrix used to add the timing constraints to the energy function, then the difficult problem—the inequality constraints as timing constraints treated by neural network—is solved. Based on the energy function, we propose a mean field annealing neural network approach for performance-driven system partitioning. In the algorithm, not only the connections between modules but also the physical structure are considered. The results show that it has more possibility to find optimal solutions.

CCACC: 7410D, 5120